



Escuela Politécnica Nacional
Facultad de Ingeniería de Sistemas

Construcción y Evolución de Software (ISWD622)

GR2SW

Grupo: EvoLogic

Proyecto [1B]

Documento de Flujo de Trabajo

Alumnos:

Molina Anael

Palma Stuart

Quillupangui Andrés

Sánchez Ariel

Profesora: Evelyn Mosquera

Fecha de entrega: 01/12/2024

FLUJO DE TRABAJO

Objetivo:

Definir el proceso y las reglas para la gestión de cambios en el código fuente utilizando las ramas main, develop y bugfix. Este flujo asegura una colaboración eficiente, control de calidad y una integración organizada en el repositorio principal.

Flujo de Trabajo

1. Estructura de Ramas

- **main:**
 - Contiene el código estable en producción.
 - Solo se actualiza con cambios completamente probados y aprobados.
- **develop:**
 - Contiene el código en desarrollo.
 - Recibe nuevas funcionalidades y correcciones antes de ser probadas para producción.
- **feature:**
 - Contiene funcionalidades nuevas, que luego serán unidas en develop.
- **bugfix:**
 - Se usa para corregir errores detectados en el código ya implementado.
 - Las correcciones se integran primero en develop y, en casos críticos, directamente en main.
- **documentos:**
 - Contiene la documentación del proyecto.

2. Creación de una Nueva Rama

- **Regla:**

Cada tarea se desarrolla en una rama independiente que se deriva de develop (para nuevas funcionalidades) o de main (para correcciones urgentes).
- **Tipos de ramas:**
 - **Feature branches (nueva funcionalidad):** feature/nombre-descriptivo
 - **Bugfix branches (corrección de errores):** bugfix/nombre-descriptivo

- **Formato de nombres:** Usar nombres descriptivos que reflejen la tarea. Ejemplo: bugfix/corrige-error-login.

3. Desarrollo en la Rama

- **Regla:**
Los cambios deben realizarse exclusivamente en la rama correspondiente.
- **Proceso:**
 - Realizar *commits* pequeños y descriptivos.
 - Seguir las guías de estilo del proyecto.
- **Validación:**
 - Ejecutar pruebas unitarias y funcionales antes de realizar un commit.

4. Sincronización con Ramas Base

- **Regla:**
Las ramas deben mantenerse actualizadas con su rama base (develop o main).
- **Proceso:**
 - Hacer *pull* regularmente desde la rama base.
 - Resolver conflictos de manera local antes de avanzar al siguiente paso.

5. Solicitud de Revisión de Código (Pull Request / Merge Request)

- **Regla:**
Cada cambio debe ser revisado antes de integrarse en las ramas develop o main.
- **Proceso:**
 1. Crear una solicitud de PR hacia la rama base (develop o main).
 2. Incluir una descripción detallada de los cambios realizados.
- **Responsables:**
 - Autor del PR.
 - Revisores asignados (mínimo dos).

6. Revisión de Código

- **Regla:**
Todos los PR deben pasar por revisión antes de ser aprobados.
- **Criterios de revisión:**
 - Correcta implementación de la funcionalidad o solución al error.
 - Cumplimiento de las guías de estilo del proyecto.
 - Cobertura adecuada de pruebas automatizadas.
- **Decisiones posibles:**
 - **Aprobado:** El PR cumple con los requisitos.
 - **Solicitar cambios:** Se deben realizar ajustes antes de proceder.

7. Integración de Cambios

- **Regla:**
Solo los PR aprobados pueden integrarse en la rama base correspondiente.
- **Proceso:**
 - **Funcionalidades nuevas:** Se integran en develop.
 - **Correcciones urgentes:** Se integran en main y se sincronizan con develop.
 - Eliminar la rama temporal después del merge.

8. Pruebas Finales e Implementación en Producción

- **Regla:**
Antes de desplegar a producción, los cambios en main deben ser probados exhaustivamente en un entorno de preproducción.
- **Proceso:**
 - Ejecutar pruebas de integración, funcionales y de aceptación en main.
 - Implementar en producción solo después de que todas las pruebas sean satisfactorias.